НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Теплоенергетичний факультет
Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:
Завідувач кафедри
_____ Олександр Коваль
«___»_____2020 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Геометричне моделювання в інформаційних системах»

спеціальності 122 «Комп'ютерні науки та інформаційні технології»

на тему: «Застосунок для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності»

Виконала:
студентка IV курсу, групи ТР-61
Фейлик Тамара Олександрівна _____

Керівник:
доцент, к.т.н.,
Демчишин А. А. _____

Рецензент:
професор, д.т.н., професор,
Несвідомін В.М. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2020 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший рівень

Напрям підготовки 122 Комп’ютерні науки та інформаційні технології

Спеціалізація Геометричне моделювання в інформаційних системах

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр Коваль
(підпис)
”___”_____2020р.

ЗАВДАННЯ
на дипломну роботу студенту
Фейлик Тамарі Олександрівні
(прізвище, ім’я, по батькові)

1. Тема роботи «Застосунок для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності»
керівник роботи Демчишин Анатолій Анатолійович, доцент, к.т.н.
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від “ ” ___ 20 р. №_____
2. Строк подання студентом роботи _____
3. Вихідні дані до роботи мова програмування JavaScript, середовище розробки Visual Studio Code.
4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) провести огляд існуючих програмних систем для полегшення процесу сервісу печатних плат, провести аналіз сучасних інструментів розробки для реалізації Web-застосунків із використанням технології доповненої реальності, розробити архітектуру програмного забезпечення із урахуванням критеріїв до її функціональних можливостей, провести дослідження залежності максимальної відстані ідентифікації маркеру суміщення 3D-моделі з її візуальним представленням на екрані пристрою від розмірів маркеру, реалізувати Web-застосунок для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності.
5. Перелік ілюстративного матеріалу 1. Об’єкт, предмет, мета роботи. 2. Завдання роботи. 3. Аналіз стану існуючих програмних систем для сервісу печатних плат. 4. Засоби розробки застосунку. 5. Функціональні можливості Web-застосунку. 6. Структура файлової бази даних. 7. Дослідження залежності максимальної відстані ідентифікації маркеру від його розмірів. 8. Приклади

роботи системи. 9. Висновки.
6. Дата видачі завдання “10” жовтня 2020 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Затвердження теми роботи	14.10.19	
2.	Вивчення та аналіз задачі	15.10.19-25.12.19	
3.	Розробка архітектури та загальної структури системи	26.12.19-26.03.20	
4.	Розробка структур окремих підсистем	27.03.20-10.04.20	
5.	Програмна реалізація системи	11.04.20-14.05.20	
6.	Оформлення пояснювальної записки	15.05.20-07.06.20	
7.	Захист програмного продукту	05.06.20	
8.	Передзахист	05.06.20	
9.	Захист	15.06.20	

Студент _____ Фейлик Т.О.
(підпис) (прізвище та ініціали,)

Керівник роботи _____ Демчишин А.А.
(підпис) (прізвище та ініціали,)

АНОТАЦІЯ

Обсяг пояснювальної записки становить 47 сторінок, міститься 19 рисунків, п’ять таблиць, 3 додатки та 7 джерел.

Метою дипломної роботи є розробка теоретичних та алгоритмічних засад візуалізації вказівок із сервісу печатних плат на основі доповненої реальності.

У процесі роботи було досліджено аналогічні програмні продукти, що полегшують сервіс печатних плат на основі доповненої реальності.

Результатом роботи є кросплатформний Web-застосунок для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності із можливостями сканування QR-коду, перегляду завантаженої 3D-моделі плати та вибору струмопровідних доріжок для відображення. Реалізована система успішно пройшла випробування на смартфоні та портативному комп’ютері.

Розроблено Web-застосунок за допомогою мови програмування JavaScript, бібліотек AR.js та Three.js у середовищі виконання Visual Studio Code.

Ключові слова: доповнена реальність, печатні плати, візуалізація, Node.js, JavaScript, Webpack, 3D-модель.

ABSTRACT

The volume of the explanatory note is 47 pages, contains 19 figures, five tables, 3 appendices and 7 sources.

The purpose of the work is to develop theoretical and algorithmic principles of visualization instructions for the PCB service on the basis of augmented reality.

The similar software products that facilitate the service of printed circuit boards based on augmented reality were investigated in the work process.

The result of the work is a cross-platform Web-application for visualization instructions for the PCB service based on augmented reality with the ability to scan the QR-code, view the loaded 3D-model of the board and select the current paths to display. The implemented system has been successfully tested on a smartphone and laptop.

The development of a Web application was implemented using the JavaScript programming language, AR.js and Three.js libraries in the Visual Studio Code environment.

Keywords: augmented reality, printed circuit boards, visualization, Node.js, JavaScript, Webpack, 3D-model.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

HTML (англ. Hypertext Markup Language) – стандартна мова розмітки гіпертекстових документів.

CSS (англ. Cascading Style Sheets) – мова таблиць стилів, що використовується для опису зовнішнього вигляду Web-сторінок.

Web (англ. World Wide Web, WWW) – всесвітня розподілена інформаційна система, що надає доступ до інформації через підключення до мережі Інтернет.

AR (англ. Augmented Reality) – це система, що у режимі реального часу розширює фізичний світ шляхом додавання до нього шарів цифрової інформації.

QR code (англ. Quick Response code) – матричний штрих-код, що містить інформацію про закріплений за ним об’єкт.

Hiro marker – матричний штрих-код для ідентифікації 3D-об’єкту.

API (англ. Application Programming Interface) – набір протоколів взаємодії, визначень підпрограм та засобів створення програмного забезпечення.

Npm (англ. Node Package Manager) – менеджер пакунків для прототипної мови програмування JavaScript.

RAM (англ. Random-Access Memory) – оперативна пам’ять.

CPU (англ. Central Processing Unit) – центральний процесор.

URL (англ. Uniform Resource Locator) – єдиний стандартизований вказівник на Web-ресурс.

ВСТУП

У XXI столітті технології стали рушійною силою глобальних змін у житті суспільства. Вплив ґаджетів та їх впровадження у повсякдення є запорукою перспективного майбутнього.

Технології, що були описані у науково-фантастичній літературі 20 років тому, вже сьогодні є реальними, наприклад, доповнена реальність.

Доповнена реальність – це технологія, що розширює фізичний світ шляхом додавання до нього шарів цифрової інформації [1]. Вона може знайти застосування у будь-якій сфері повсякденного життя. Вже сьогодні люди мають змогу за допомогою звичайного смартфону або планшету із камерою розставити нові меблі у своєму домі та подивитись, чи пасуватимуть вони до інтер'єру, або ж вивчати анатомію людини із можливістю деталізації кожного органу.

Для більшості комп'ютерних сервісів як ніколи актуальною є проблема ремонту печатних плат. Велика кількість маленьких деталей та складність струмопровідних доріжок створює значні труднощі при їх обслуговуванні. Застосунок для візуалізації вказівок із сервісу печатних плат значно полегшує процес їх ремонту завдяки наочному відображенню всіх її складових. Окрім того, зображення деталізованої 3D-моделі пристрою робить навчання нових спеціалістів із комп'ютерного сервісу значно ефективнішим та швидшим.

У контексті даної роботи під "вказівками із сервісу печатних плат" маються на увазі візуальні інструкцій на екрані пристрою користувача, а під сервісом – процес ремонту та обслуговування струмопровідних доріжок, що з'єднують компоненти печатної плати.

Об'єктом даної роботи є розробка програмного забезпечення для сервісу печатних плат.

Предметом роботи є розробка Web-застосунку із використанням технології доповненої реальності.

Метою даної дипломної роботи є розробка теоретичних та алгоритмічних засад візуалізації вказівок із сервісу печатних плат на основі доповненої реальності.

Для досягнення мети було поставлено наступні задачі, що визначили структуру та логіку роботи:

1. Проаналізувати існуючі програмні системи для вказівок із сервісу печатних плат.
2. Проаналізувати сучасні інструменти для розробки Web-застосунків із використанням технології доповненої реальності.
3. Розробити архітектуру програмного забезпечення із урахуванням критеріїв до її функціональних можливостей та провести дослідження залежності максимальної відстані ідентифікації маркеру суміщення 3D-моделі з її візуальним представленням на екрані пристрою від розмірів маркеру.
4. Реалізувати Web-застосунок для вказівок із сервісу печатних плат, що дасть змогу спростити процес ремонту печатних плат та пришвидшити навчання нових спеціалістів.

Роботу виконано в контексті НДР 0119U103633 “Аналіз і візуалізація геометричних та геоінформаційних даних”.

Зміст розділів даної наукової роботи наступний:

Перший розділ роботи присвячений аналізу програмних систем для сервісу печатних плат.

У другому розділі обґрунтовується вибір засобів реалізації Web-застосунку.

У третьому розділі роботи описується архітектура програмного забезпечення та дослідження залежності відстані ідентифікації маркеру від його розмірів.

У четвертому розділі наведено методику роботи користувача із застосунком.

1. АНАЛІЗ СТАНУ ПРОГРАМНИХ СИСТЕМ ДЛЯ СЕРВІСУ ПЕЧАТНИХ ПЛАТ

1.1. Постановка задачі

Мета реалізації програмної системи – зробити процес ремонтного обслуговування печатних плат та навчання нових спеціалістів із комп'ютерного сервісу більш простим та швидким. Обрано застосунок для візуалізації вказівок із сервісу печатних плат, використовуючи можливості технології доповненої реальності.

Призначенням створеного програмного забезпечення є наочне відображення компонентів печатної плати та відповідних струмопровідних доріжок, що їх з'єднують. Потенційними користувачами даного застосунку стануть власники комп'ютерних ремонтних сервісів та особи, що бажають освоїти професію сервісного інженера із ремонту комп'ютерної техніки.

Розроблений продукт має відповідати наступним критеріям:

- наявність адаптивного інтерфейсу для зручності взаємодії користувача із системою на різних пристроях із камерою, а саме портативному комп'ютері, персональному комп'ютері, смартфоні та планшеті;
- сумісність щонайменше з двома сучасними Web-браузерами (Google Chrome та Mozilla Firefox).

Програмний продукт повинен мати наступні функції:

- можливість зчитування та розпізнавання QR-кодів;
- можливість зчитування та розпізнавання Hiro-маркерів;
- відображення 3D-моделі печатної плати на екрані пристрою із камерою у режимі реального часу;

– вибір та відображення струмопровідних доріжок, що з’єднують певні компоненти плати на 3D-моделі.

Web-застосунок має складатися із трьох головних модулів:

– QR-сканеру;
– AR-модулю;
– бази даних моделей.

Архітектура програми має пов’язувати всі три модулі між собою. Сканер QR-коду повинен отримувати інформацію про модель та надсилати запит про пошук до бази даних, а AR-модуль отримувати результат та відображати відповідну 3D-модель у режимі реального часу.

Для розробки застосунку необхідно використати бібліотеки Three.js та AR.js, технології HTML5, CSS3, JavaScript, Node.js та інструмент для компіляції модулів Webpack. 3D-модель має відображатися на екрані пристрою з камерою за допомогою технології розпізнавання Hiro-маркерів, що містять у собі зашифровану інформацію про об’єкт. Для кожної печатної плати необхідно згенерувати унікальний QR-код та розмістити безпосередньо на ній.

1.2. Огляд існуючих програмних систем

Задача спрощення процесу сервісу комп’ютерних складових на сьогодні є досить актуальною, адже зі збільшенням кількості пристроїв росте і попит на їх налагодження.

Найбільш поширеними серед дефектів печатних плат є:

– дефекти металізації;
– дефекти ізоляційного матеріалу;
– дефекти фінішних покриттів;
– дефекти струмопровідних доріжок [2].

Серед описаних вище дефектів найважче налагоджувати струмопровідні доріжки, адже складність їх структури значно збільшує терміни обслуговування плати. Вирішення даної проблеми дозволить зробити процес ремонту компонентів комп’ютерів менш часо-затратним.

На ринку інформаційних технологій вже існують наступні програмні продукти, що повністю або частково реалізують розв'язок даної задачі:

1. InspectAR.
2. DebuggAR.

Вищеперераховані застосунки мають наступні недоліки: реалізація лише бета-версії, дорога вартість річної підписки та високий поріг входження для користування.

До переваг існуючих програмних продуктів можна віднести використання новітніх технологій, а також взаємодію із впливовими компаніями у галузі 3D-моделювання та прототипування.

1.3. Застосунок InspectAR

InspectAR – це сучасний програмний продукт, створений компанією InspectAR Augmented Interfaces Inc. для ознайомлення, налагодження та збірки печатних плат на основі доповненої реальності [3].

Застосунок надає наступний функціонал:

1. Дослідження компонентів печатних плат.
2. Фільтрація компонентів за типами.
3. Пошук компонентів на платі.
4. Доступ до існуючої документації.
5. Створення та редагування власної документації.
6. Перевірка розміщення компонентів на платі.

Переваги продукту:

– швидкодія;
– кросплатформність;
– можливість взаємодії із іншими користувачами;
– доступ до документації;
– візуалізація сигналів.

Недоліки продукту:

– висока вартість річної підписки для професійного використання, що становить 149 доларів на рік (рисунок 1.1);
– складність впровадження командної розробки;
– високий поріг входження для користування (рисунок 1.2).

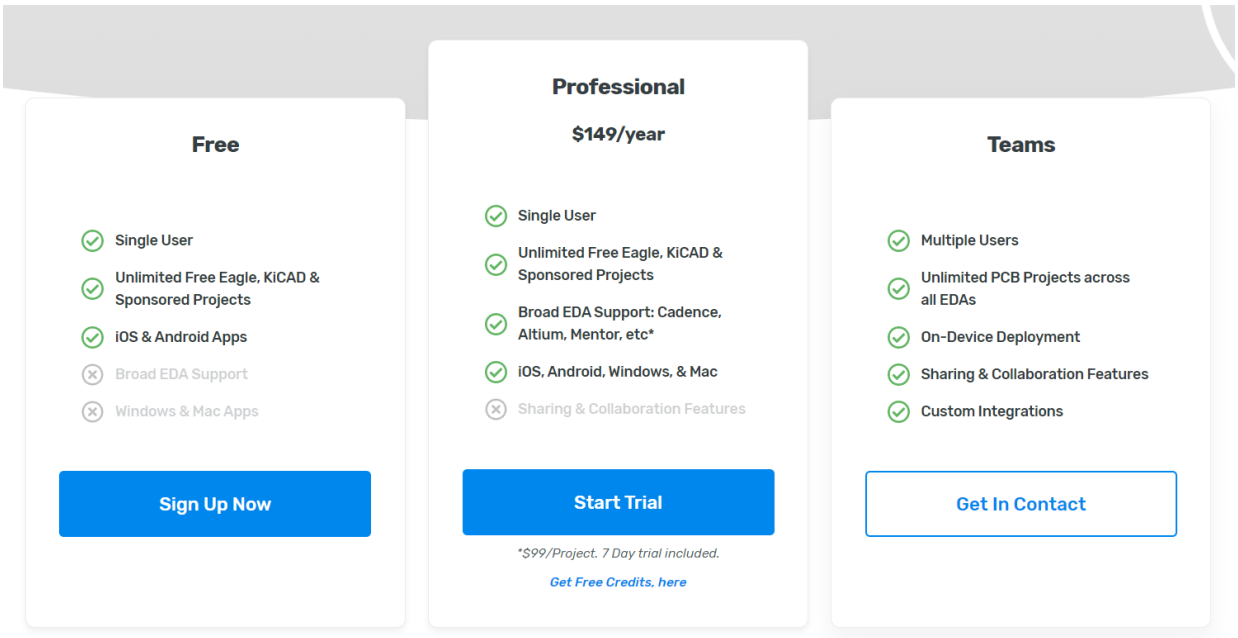

Рисунок 1.1 – Вартість річної підписки для професійного користування застосунком InspectAR

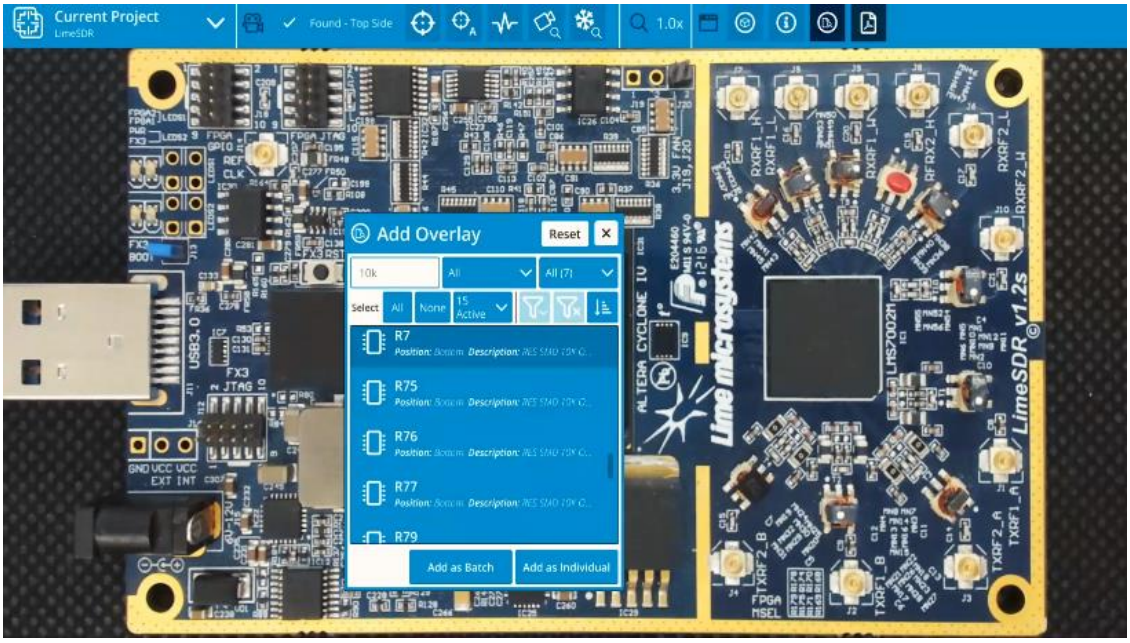

Рисунок 1.2 – Користувальницький інтерфейс Web-застосунку InspectAR

Розробка компанії InspectAR Augmented Interfaces Inc. надає широкий функціонал для тестування, налагодження та збирання печатних плат, але має обмежену цільову аудиторію.

1.4. Застосунок DebuggAR

DebuggAR – Web-застосунок на основі доповненої реальності для ремонту, модернізації та збирання печатних плат.

Функціональні можливості розробка компанії DebuggAR:

1. Підсвітка компонентів плати у режимі реального часу із використанням технології доповненої реальності.
2. Можливість багатошарового менеджменту.
3. Створення та завантаження власної схеми плати.
4. Аналіз сигналів, шарів та компонентів (рисунок 1.3).

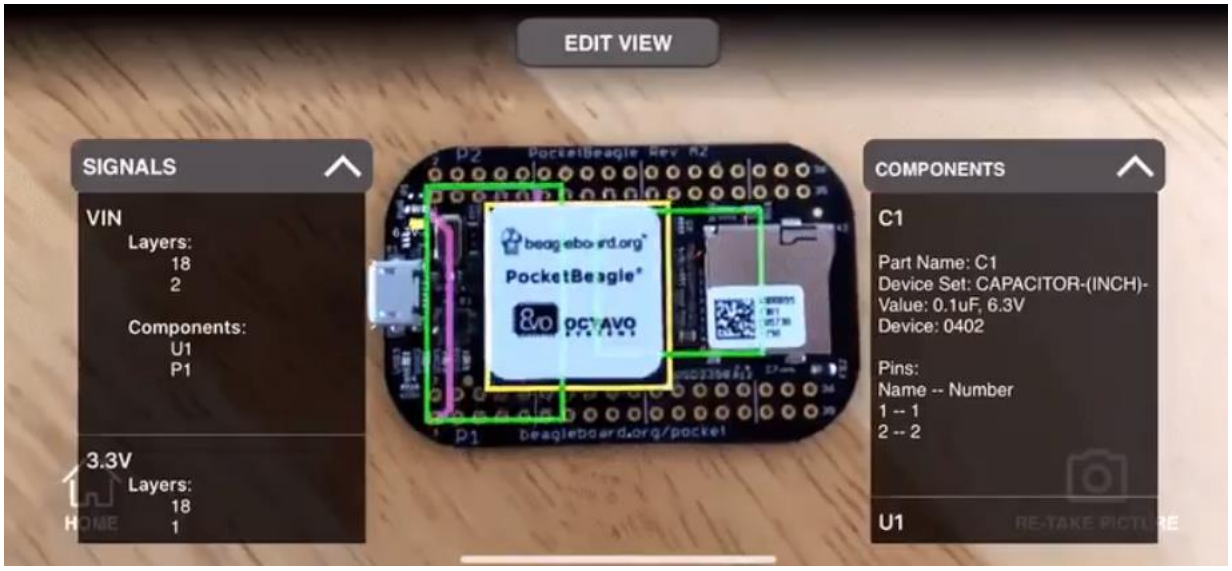

Рисунок 1.3 – Візуалізація та аналіз сигналів за допомогою застосунку DebuggAR

Головними перевагами застосунку DebuggAR є:

- можливість роботи із кількома шарами печатної плати;
- розробка та тестування власних схем.

Серед недоліків слід вказати відсутність підтримки роботи програми на різних платформах та наявність лише нестабільної бета-версії для тестування.

DebuggAR є перспективним продуктом, що потребує завершення етапу тестування та реалізації програми для повноцінного використання на різноманітних платформах.

Висновки до розділу

У даному розділі було проведено аналіз існуючих програмних систем для оптимізації сервісу печатних плат із визначенням їх переваг та недоліків та поставлено задачу розробки програмного забезпечення для наочного відображення компонентів печатної плати та відповідних струмопровідних доріжок, що їх з’єднують.

2. ОБҐРУНТУВАННЯ ВИБОРУ ЗАСОБІВ РЕАЛІЗАЦІЇ ЗАСТОСУНКУ

Web-розробка є одним із найбільш популярних напрямів сучасної IT-галузі, що спричинило появу нових інструментів для розробки програмного забезпечення. Вибір оптимальних технологій гарантує можливість розробки кросплатформного, стабільного та якісного Web-продукту.

Для реалізації застосунку було використано набір технологій, зображених на рисунку 2.1.

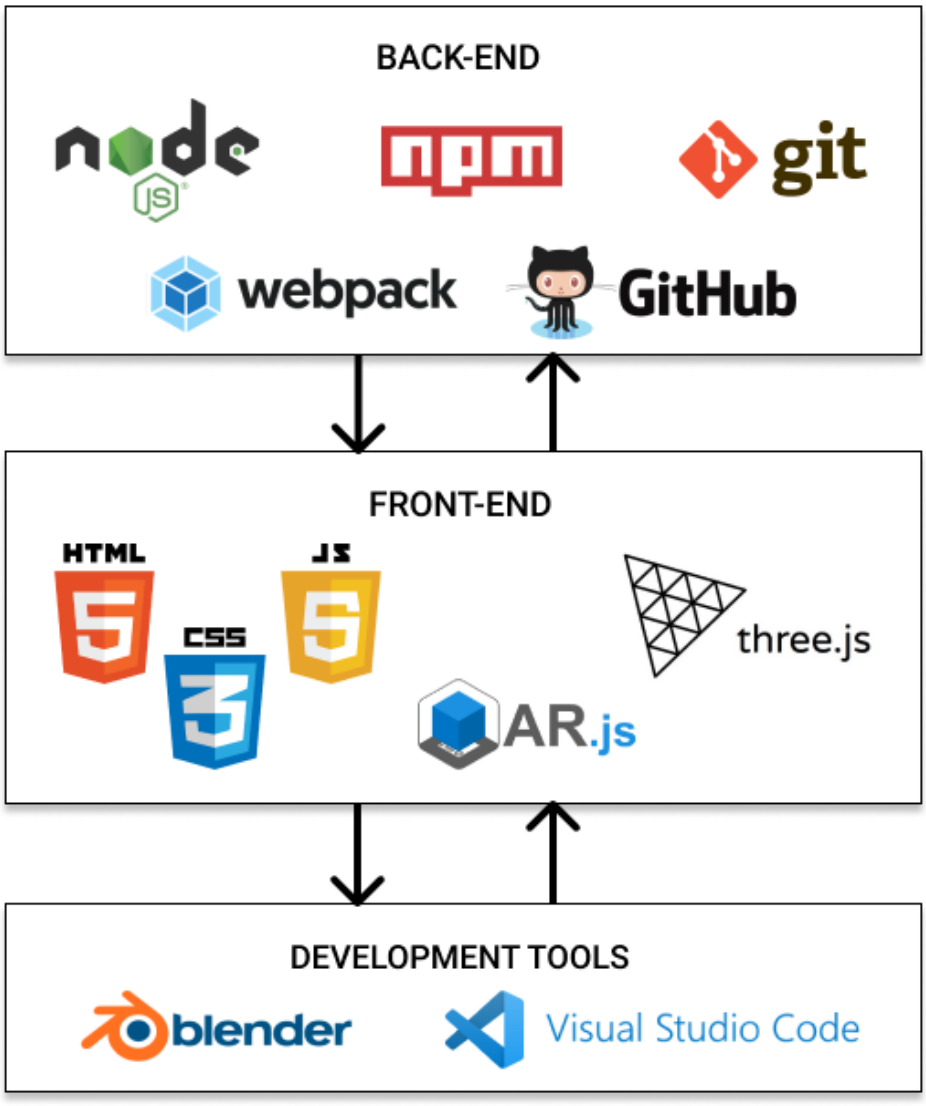

Рисунок 2.1 – Технології для розробки програмної системи

Перелік використаних інструментів для реалізації задачі візуалізації вказівок із сервісу печатних плат наступний:

- мова програмування Node.js для побудови масштабованих сервісних застосунків;
- npm – менеджер для управління пакетами Node.js;
- система контролю версій Git;
- інструмент для компіляції модулів Webpack;
- Web-сервіс для хостингу та спільної розробки проектів GitHub;
- стандартна мова розмітки гіпертексту HTML5;
- мова таблиць стилів CSS;
- динамічна об’єктно-орієнтована прототипна мова програмування JavaScript;
- бібліотеки Three.js та AR.js;
- засіб для створення, редагування та налагодження Web-застосунків Visual Studio Code;
- відкрите програмне забезпечення для створення тривимірної комп’ютерної графіки Blender.

2.1. Засоби реалізації front-end частини Web-застосунку

Для реалізації клієнтської частини Web-застосунку було використано наступні технології:

1. Мови розмітки гіпертексту HTML та таблиць стилів CSS.
2. Прототипну мову програмування JavaScript.
3. Бібліотеки Three.js та AR.js.

2.1.1. Мова розмітки гіпертексту HTML

HTML (англ. HyperText Markup Language) – це мова розмітки гіпертексту, що знайшла широке застосування серед Web-застосунків. Псевдомова визначає логічну структуру сторінок Web-браузеру.

Front-end частина даного застосунку була реалізована із використанням мови розмітки гіпертексту HTML5 (англ. HyperText Markup Language, version 5). HTML5 – це відкрита платформа для створення Web-застосунків, що передбачають використання анімацій, графіки, відео- та аудіо- контенту. Псевдомова п’ятої версії є удосконаленням попередньої HTML.

Переваги використання HTML5 наступні:

1. Простота структури елементів на сторінці, що оптимізує процес створення та налагодження програмного коду.
2. Інтеграція із різноманітними інтерфейсами.
3. Наявність вбудованих елементів для управління медіа-контентом.

2.1.2. Мова таблиць стилів CSS

CSS (англ. Cascading Style Sheets) – це формальна мова для дизайну зовнішнього вигляду Web-сторінок, що були створені за допомогою мов розмітки гіпертексту.

Використання даної технології дозволяє:

- відокремлювати зміст Web-документу від його оформлення, що сприяє спрощенню його структури;
- керувати розміщенням та стилізацією елементів Web-сторінки;
- створювати адаптивний інтерфейс користувача для коректного відображення Web-сторінок на різних платформах.

2.1.3. Мова програмування JavaScript

JavaScript – скриптова мова програмування для розробки динамічного контенту Web-сторінок. Дана мова підтримує функціональний, подійно-орієнтований та імперативний стилі програмування. Автоматичне управління пам’яттю, підтримка динамічної та слабкої типізації, прототипне програмування – головні архітектурні особливості JavaScript.

Область використання даної технології досить широка:

- клієнтська частина Web-застосунків;
- реалізація серверних застосунків;
- розробка мобільних застосунків;
- реалізація браузерних операційних систем;
- створення невеликих програм-букмарклетів, для розширення функціональних можливостей браузерів;
- розробка допоміжних програм-віджетів;
- створення прикладного програмного забезпечення.

Переваги та недоліки є у будь-якої мови програмування, і JavaScript не є виключенням. Дана технологія не завжди підтримується окремими Web-браузерами чи пристроями. Окрім того, JavaScript вразлива для шкідливого програмного коду та може бути використана для його створення на пристрої користувача.

Переваги даної технологій наступні:

1. Низький поріг входження для користування.
2. Можливість створення унікальних інтерактивних Web-застосунків.
3. Простота виявлення помилок у вихідному коді.
4. Можливість розробки кросплатформних застосунків.

2.1.4. Бібліотеки Three.js та AR.js

Three.js – бібліотека, що містить набір готових класів для створення та відображення якісної анімованої комп’ютерної графіки на Web-сторінках. Весь вихідний код технології міститься у відкритому доступі на ресурсі GitHub. Three.js є бібліотекою високого рівня, що дозволяє відображати тривимірні анімації у Web-браузері без використання спеціальних додатків чи плагінів.

Головними перевагами бібліотеки є:

- легкість вивчення та користування;
- кросплатформність;
- широкі функціональні можливості для маніпулювання 3D-графікою;
- наявність якісної документації.

Серед недоліків бібліотеки слід зазначити підтримку роботи лише у сучасних браузерах та мобільних пристроях.

AR.js – бібліотека JavaScript, що дозволяє використовувати можливості доповненої реальності на основі маркерів або геолокації у сучасних Web-застосунках. Як і Three.js, технологія міститься у відкритому доступі на ресурсі GitHub.

Головні переваги AR.js:

- кросплатформність;
- висока продуктивність до 60 кадрів у секунду навіть на старих пристроях;
- простота у підключенні;
- використання браузерних API, доступних на комп‘ютерах та мобільних девайсах.

Недоліками AR.js є неповнота технічної документації та проблеми роботи бібліотеки на багатокамерних пристроях у браузері Google Chrome.

2.2. Засоби реалізації back-end частини Web-застосунку

Для реалізації програмно-апаратної частини Web-застосунку було використано наступні технології:

1. Середовище виконання Node.js.
2. Інструмент Webpack.

2.2.1. Середовище виконання Node.js

Node.js – асинхронна технологія для побудови масштабованих мережевих застосунків, реалізованих на мові програмування JavaScript [4]. Головна ідея середовища виконання Node.js полягає у використанні неблокуючого подійно-орієнтованого введення/виведення, задля ефективності при роботі із програмами, що обробляють великі об’єми даних у реальному часі на розподілених пристроях [5].

Технологія широко використовується для реалізації серверної частини застосунків. Гнучкість, наявність великої кількості зовнішніх бібліотек і готових модулів, менеджер для управління пакетами npm є головними перевагами середовища виконання Node.js.

2.2.2. Інструмент Webpack

Інструмент Webpack дозволяє компілювати модулі, створені мовою програмування JavaScript у цілісний JS-файл. Схема роботи збірника модулів представлена на рисунку 2.2.

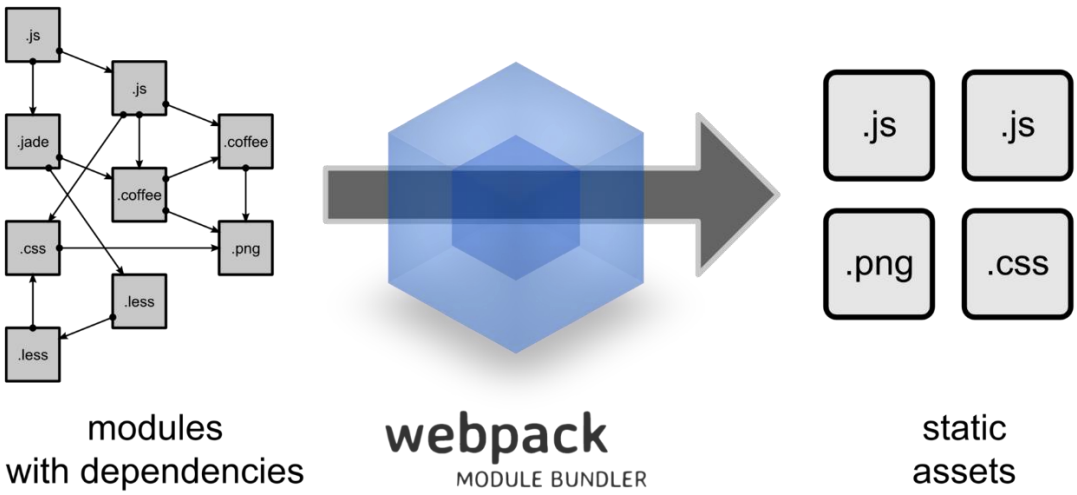

Рисунок 2.2 – Схема роботи збірника модулів Webpack

Webpack аналізує залежності між модулями та бібліотеками у програмному коді та генерує із них файли-бандлери (пакети чи вузли). Така функціональність збірника модулів дозволяє Web-розробникам краще організовувати ресурси для функціонування системи.

2.3. Середовище розробки Visual Studio Code

Visual Studio Code – це насамперед редактор, що поєднує у собі усі необхідні функції для високопродуктивного редагування вихідного коду програми. Середовище розробки розповсюджується безкоштовно та доступне у версіях для платформ Windows, Linux та OS X [6].

Visual Studio Code підтримує багато сучасних мов програмування, зокрема:

- Python;
- C / C++ / C#;
- Java;
- Go;

- PHP;
- Ruby;
- CSS;
- HTML;
- JavaScript;
- JSON;
- PowerShell;
- та інші.

Головні функціональні можливості даного середовища розробки наступні:

1. IntelliSense – система для автодоповнення вихідного коду програми із використанням можливостей штучного інтелекту. На основі імпортованих модулів, встановлених розширень, типів змінних та визначень функції технологія пропонує варіанти доповнення до коду програми у процесі його створення.
2. Інтегрований налагоджувач для автоматизації процесу пошуку помилок у вихідному коді.
3. Зручний та сучасний інтерфейс. Користувач має змогу самостійно налаштувати зовнішній вигляд редактору, визначити власні гарячі клавіші, конфігураційні файли, панелі інструментів, розмір шрифтів, вікон та багато іншого (рисунок 2.3).
4. Інтерфейс для роботи із системою контролю версій Git.
5. Підтримка сотні мов програмування. У редактор Visual Studio Code за замовчуванням вже встановлені мови JavaScript, TypeScript, CSS та HTML. Для роботи з іншими технологіями необхідно встановити плагіни через інтерфейс користувача або ж за допомогою платформи VS Code Marketplace.
6. Кросплатформність. Середовище розробки доступне для використання на базі операційних систем Windows, Linux та OS X.

7. Доступність. Visual Studio Code – безкоштовне програмне забезпечення із відкритим вихідним кодом.

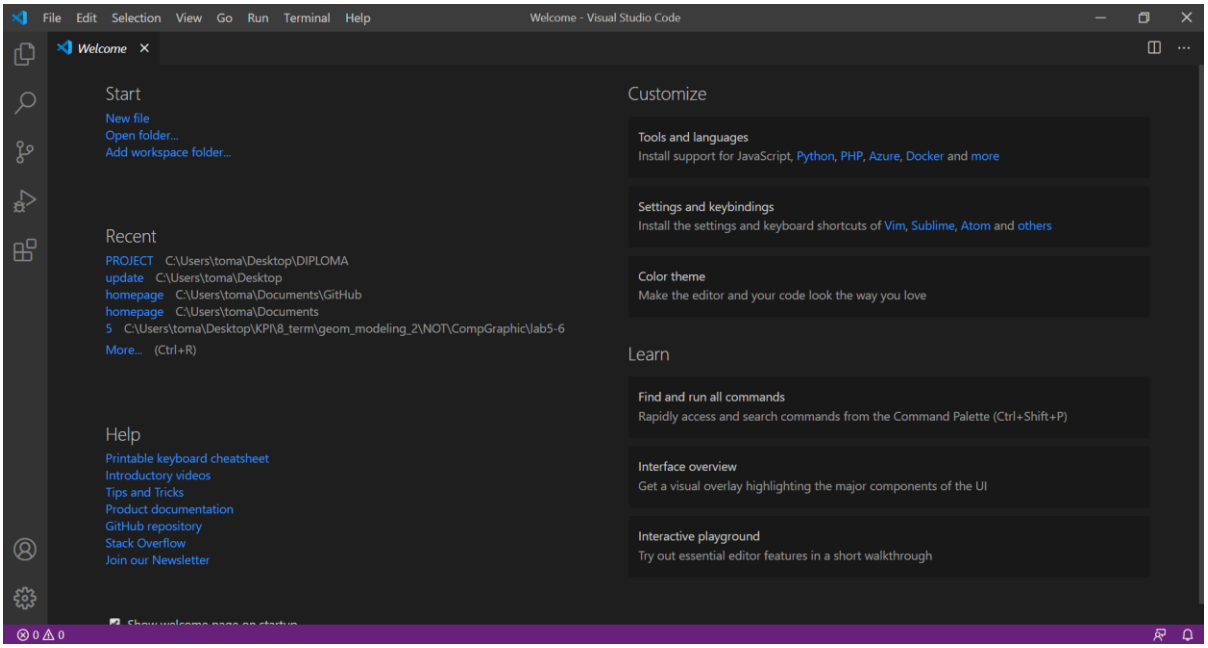

Рисунок 2.3 – Користувальницький інтерфейс Visual Studio Code

Серед недоліків Visual Studio Code слід вказати високий рівень витрат енергоресурсів пристрою.

2.4. Програмне забезпечення Blender

Blender – програмне забезпечення для створення двовимірної та тривимірної комп’ютерної графіки.

Можливості даного пакету досить широкі:

- засоби моделювання;
- цифрова скульптура;
- рендеринг;
- анімація;

– монтаж та обробка відео;

– симуляція та створення 2D-анімацій.

Blender знайшов широке застосування як серед професійних 3D-дизайнерів, так і серед новачків галузі.

Вхідний поріг для користуванням даним програмним забезпеченням досить високий. Це пов’язано із складністю графічного інтерфейсу та наявністю великої кількості гарячих клавіш майже для кожної функції. Користувальницький інтерфейс Blender представлений на рисунку 2.4.

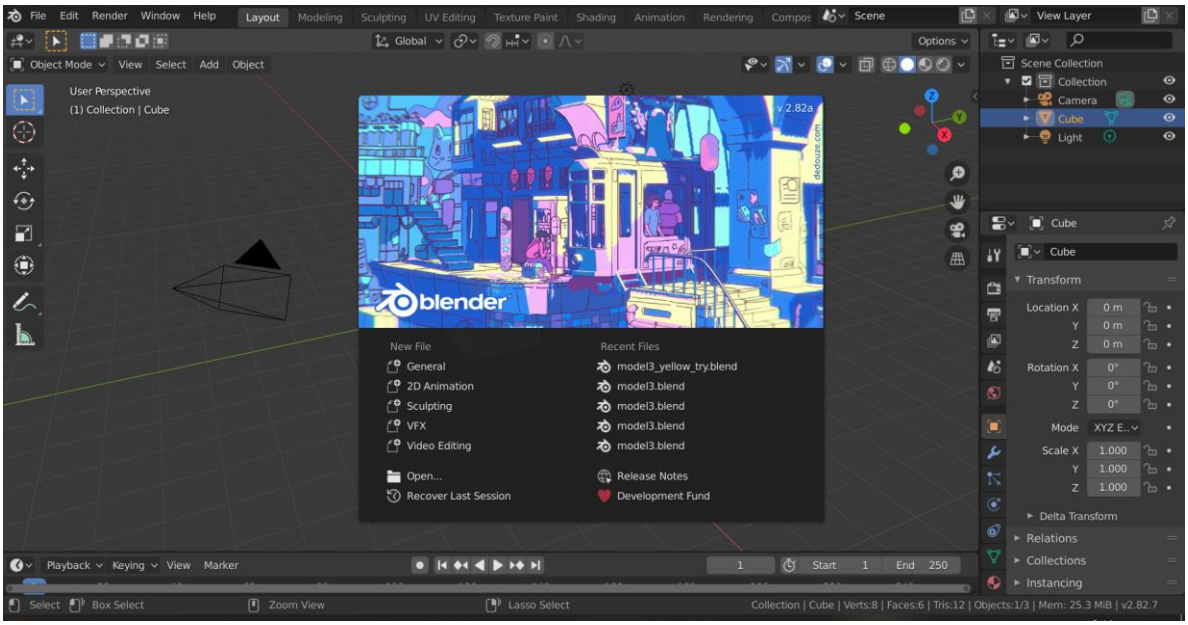

Рисунок 2.4 – Користувальницький інтерфейс програмного забезпечення Blender

Два основних режими редагування об’єктів дозволяють зручно маніпулювати їх фактичними та фізичними даними. У “Edit mode” (режим редагування) можна керувати формою та розмірами об’єктів, а у “Object mode” (об’єктний режим) – фізичними властивостями об’єктів. Автоматичне збереження та резервне копіювання проектів під час роботи програми є великою перевагою даного пакету, адже дозволяє залишити дані неушкодженими при непередбачуваних ситуаціях.

Blender є безкоштовним програмним забезпеченням, що робить його доступним для всіх охочих.

Висновки до розділу

Правильний вибір засобів програмної реалізації гарантує можливість створення кросплатформного, стабільного та якісного Web-застосунку. Окрім того, інструменти розробки помітно впливають на продуктивність системи.

У даному розділі було проведено огляд засобів програмної реалізації для розробки Web-застосунку із визначенням їх головних переваг та недоліків.

Найбільш відповідними засобами для створення front-end частини продукту є JavaScript, HTML5, CSS та бібліотеки Three.js і AR.js, а для back-end частини – Node.js, Webpack, Git та GitHub. Як середовище розробки вихідного коду було обрано Visual Studio Code, а для створення 3D-моделі печатної плати – програмне забезпечення Blender.

3. АРХІТЕКТУРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Трендом останніх років є розробка Single Page Application (односторінковий застосунок). Найбільш зручним підходом для реалізації подібних проектів є використання фреймворків із дотриманням атомарного підходу до проектування компонентів.

У даній науковій роботі користувальницький інтерфейс грає меншу роль за швидкодію та кросплатформність, тому було вирішено відмовитись від фреймворків і розробити застосунок, використовуючи найновіші специфікації JavaScript. Для зручності розробки, тестування і масштабування продукту всі логічні блоки програми поділено на модулі, які у подальшому інструмент Webpack обробляє та збирає в єдиний односторінковий Web-застосунок.

Для злагодженої роботи усіх компонентів системи швидкі утиліти розроблено у вигляді чистих функцій, а частини програми, що працюють із браузерними технологіями, реалізовано у вигляді класів.

3.1. Опис функціональних можливостей

Найкраще функціональні можливості системи демонструє діаграма прецедентів. Діаграма прецедентів (англ. use case diagram) – це опис послідовності дій, що виконуються системою і приносять вагомий результат конкретному діючому обличчю (актору). Такі діаграми використовуються для структурування поведінкових сутностей моделі [7].

Use case діаграма являє собою граф, що утворюється шляхом поєднання множин акторів, прецедентів (варіантів використання), асоціацій, відношень та границь системи.

Застосунок має лише одного актора – користувача системи.

Прецеденти реалізованого Web-застосунку наступні:

- сканування QR-коду;
- перегляд завантаженої 3D-моделі плати;
- вибір струмопровідних доріжок для відображення на 3D-моделі.

Use case діаграма Web-застосунку представлена на рисунку 3.1.

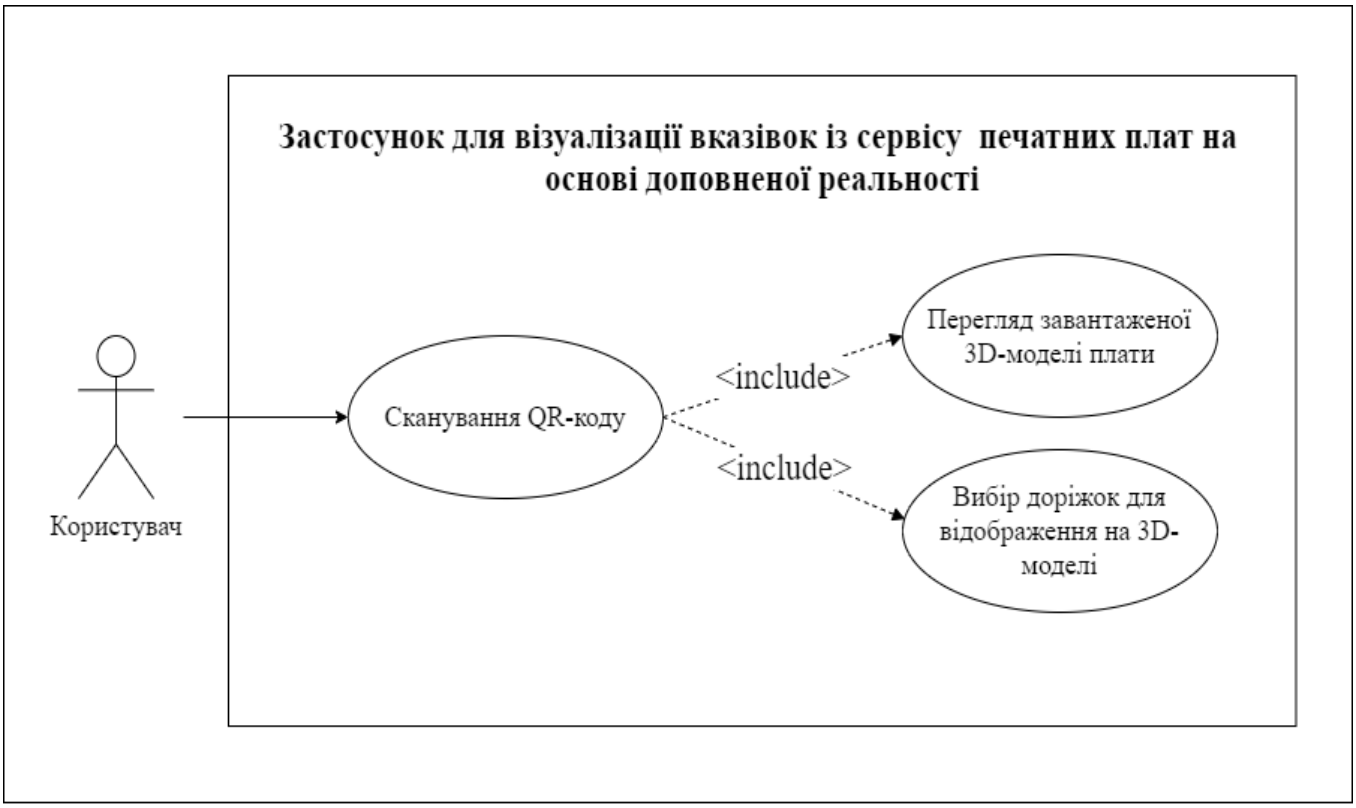

Рисунок 3.1 – Use case діаграма Web-застосунку

Користувач має можливість відсканувати QR-код, розміщений безпосередньо на печатній платі для її ідентифікації у системі. Функції перегляду завантаженої 3D-моделі та вибору струмопровідних доріжок для відображення на 3D-моделі доступні лише після сканування QR-коду.

3.2. Схема роботи Web-застосунку

Зважаючи на вимоги до функціональності програмного забезпечення, було вирішено розробити систему із модульною архітектурою. Схему роботи Web-застосунку представлено на рисунку 3.2.

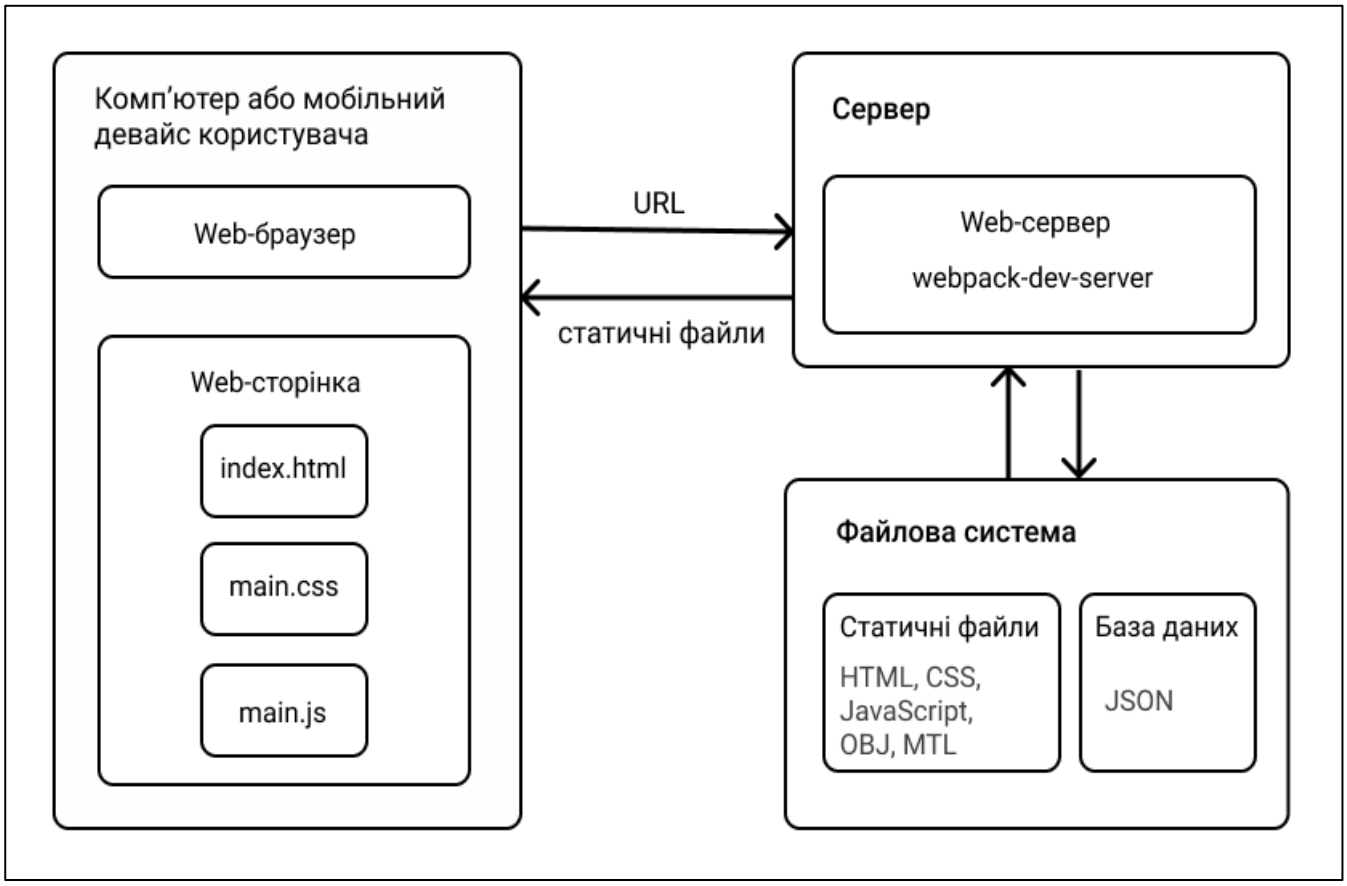

Рисунок 3.2 – Схема роботи Web-застосунку

Логічним центром усієї системи є webpack-dev-server, що обробляє та збирає всі логічні блоки програми у єдиний односторінковий застосунок.

Три головні модулі програми наступні:

1. Сканер QR-коду.
2. Файлова база даних моделей.
3. AR-модуль.

Схема взаємодії головних модулів застосунку представлена на рисунку 3.3.

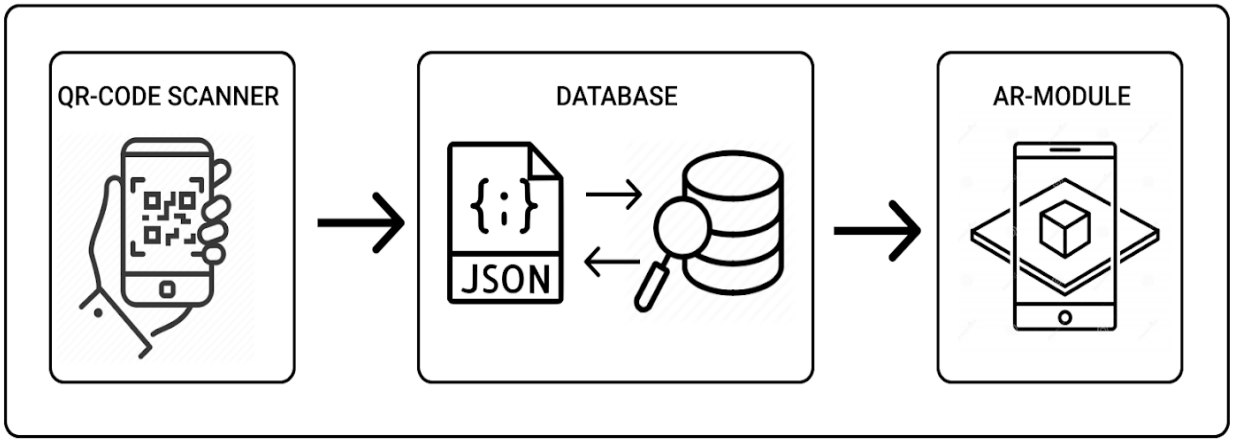

Рисунок 3.3 – Схема взаємодії головних модулів Web-застосунку

Всі три модулі взаємопов’язані. За допомогою камери пристрою QR-модуль розпізнає QR-код на печатній платі та ідентифікує її. Далі QR-модуль надсилає пошуковий запит до бази даних. AR-модуль у свою чергу отримує результат пошуку та розпізнає у відеопотоці Hiro-маркер. Внаслідок успішної ідентифікації маркера модуль доповненої реальності відображає відповідну 3D-модель безпосередньо над маркером. Таким чином, у режимі реального часу на екрані пристрою з’являється 3D-модель печатної плати.

3.2.1. QR-модуль

QR-модуль виконує наступні функції:

- зчитування та розпізнавання QR-кодів;
- відправка запит про пошук до файлової бази даних;
- повернення результату пошукового запиту.

3.2.2. Файлова база даних

Під час постановки задачі було вирішено реалізувати базу даних у вигляді файлу details.js, що містить у собі детальну інформацію про печатні плати (таблиця 3.1).

Таблиця 3.1

Структура файлової бази даних Web-застосунку

Ім’я змінної		**Тип змінної**	**Опис**
id		string	ідентифікатор плати
name		string	назва плати
scale	x	number	x координата масштабу 3D-моделі
	y	number	y координата масштабу 3D-моделі
	z	number	z координата масштабу 3D-моделі
position	x	number	x координата позиції 3D-моделі відносно Hiro-маркера
	y	number	y координата позиції 3D-моделі відносно Hiro-маркера
	z	number	z координата позиції 3D-моделі відносно Hiro-маркера
levels	name	string	ім’я струмопровідної доріжки
	file	string	шлях до файлу із розміщенням 3D-об’єкту

Кожна печатна плата має:

- унікальний ідентифікатор “id”;
- ім’я “name”;
- масштаб 3D-моделі “scale” із координатами x, y, z;
- позицію 3D-моделі відносно Hiro-маркера “position” із координатами x, y, z;
- кортеж “levels” із іменами струмопровідних доріжок “name” та інформацією про шлях до файлу із розміщенням 3D-об’єкту і його текстури “file”.

3.2.3. AR-модуль

AR-модуль виконує наступні функції:

- завантаження 3D-моделей печатної плати та її струмопровідних доріжок;
- ініціалізація головного меню застосунку для вибору доріжок;
- зчитування та розпізнавання Hiro-маркерів;
- відображення 3D-моделей печатної плати та її струмопровідних доріжок.

3.3. Створення 3D-моделі печатної плати

Можливості бібліотеки AR.js дозволяють відображати різноманітні 3D-об’єкти, створені за допомогою бібліотеки Three.js або ж стороннього програмного забезпечення. Із огляду на наявність на печатній платі Foxconn K8S755M-6ELRS великої кількості компонентів та струмопровідних доріжок, було вирішено створити її 3D-прототип програмним забезпеченням Blender.

На основі фізичної плати та документації до неї було створено її 3D-модель (рисунок 3.4).

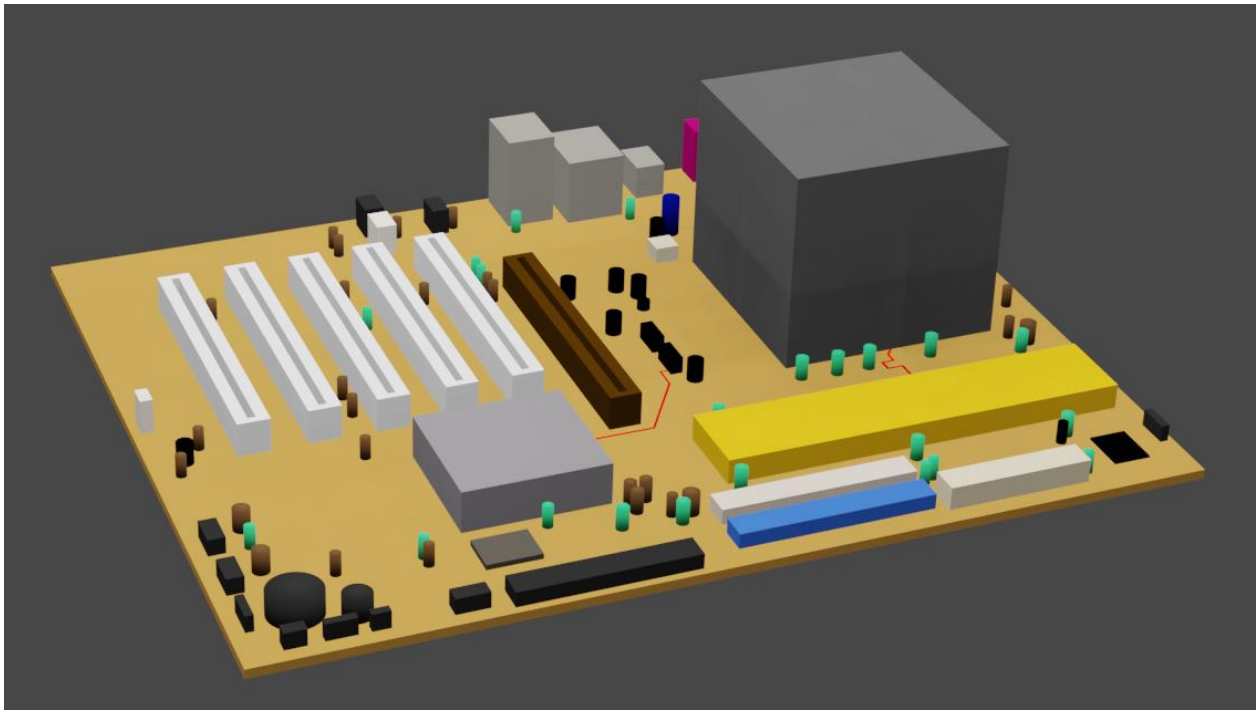

Рисунок 3.4 – 3D-модель печатної плати Foxconn K8S755M-6ELRS

Струмопровідні доріжки було розроблено як окремі 3D-об’єкти, що накладаються на базову модель.

3.4. Дослідження якості трекінгу Hiro-маркерів

Якість трекінгу Hiro-маркерів значно впливає на зручність користування Web-застосунком на основі технології доповненої реальності. Якщо трекінг маркеру не стабільний, 3D-об’єкт не буде коректно відображатись на екрані пристрою. Постійне “мигання” деталі негативно впливатиме на досвід роботи користувача із реалізованою системою.

Було проведено дослідження залежності максимальної відстані ідентифікації маркеру суміщення 3D-моделі з її візуальним представленням на екрані пристрою від розмірів маркеру. Результати дослідження наведено у таблиці 3.2.

Таблиця 3.2

Якість трекінгу Hiro-маркерів залежно від їх розмірів

Відстань від Hiro-маркера до камери (см)	Розмір Hiro-маркеру (см)							
	1×1	2×2	3×3	4×4	5×5	6×6	7×7	8×8
10								
15								
20								
25								
30								
35								
40								
45								
50								
55								
60								
65								
70								
75								
80								
85								
90								
95								
100								

Таблиця 3.3

Легенда таблиці 3.2

	Маркер розпізнається стабільно
	Маркер розпізнається, але нестабільно
	Маркер не розпізнається

Із таблиці 3.2 видно, що область стабільного трекінгу маркеру знаходиться на діагоналі матриці (поля зеленого кольору). Області нестабільного трекінгу знаходяться вище та нижче головної діагоналі (поля жовтого кольору). Область, де

трекінг відсутній, знаходиться над областями нестабільного трекінгу (поля червоного кольору).

Висновки до розділу

У даному розділі роботи описано:

- функціональні можливості реалізованої системи;
- особливості її модульної архітектури;
- взаємодію між головними модулями Web-застосунку.

Окрім того, було проведено дослідження залежності максимальної відстані ідентифікації маркеру від розмірів маркеру.

4. МЕТОДИКА РОБОТИ КОРИСТУВАЧА ІЗ WEB-ЗАСТОСУНКОМ

Для роботи із Web-застосунком користувачу необхідно мати пристрій із камерою (персональний або портативний комп’ютер, смартфон, планшет) та стабільне підключення до мережі Інтернет.

Розроблений продукт кросплатформний та коректно працює у сучасних Web-браузерах Google Chrome версії 81.0.4044.138 і вище та Mozilla Firefox версії 76.0.1 і вище.

4.1. Інсталяція та системні вимоги

Web-застосунок для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності не потребує встановлення на пристрій користувача, адже є Web-системою.

Доступ до програмного продукту здійснюється за URL-посиланням.

Обов’язкові вимоги для безперебійної роботи застосунку наступні:

1. Швидке з’єднання із мережею Internet (не менше 5Мбіт/с).
2. Наявність сучасного Web-браузера Google Chrome версії 81.0.4044.138 і вище або Mozilla Firefox версії 76.0.1 і вище.
3. Пристрій із камерою (смартфон, планшет, портативний або персональний комп’ютер).
4. Об’єм оперативної пам’яті пристрою не менше 2GB.

Тестування застосунку було проведено на портативному комп’ютері Lenovo ThinkPad T450s та на смартфоні Xiaomi Redmi Note 8T. Web-система успішно пройшла випробування на вищеперерахованих пристроях.

Технічні характеристики пристроїв, на яких було проведено тестування Web-застосунку наведено у таблицях 4.1 та 4.2 відповідно.

Таблиця 4.1

Технічні характеристики пристрою Lenovo ThinkPad T450s

Характеристики	**Lenovo ThinkPad T450s**
Процесор	Intel(R) Core(TM) i5-5330U
RAM (оперативна пам’ять)	12GB
Операційна система	Windows10 x 64
CPU (центральний процесор)	частота 2.30Ghz
Графічний чип	Intel(R) HD Graphics 5500
Web-камера	720p

Таблиця 4.2

Технічні характеристики пристрою Xiaomi Redmi Note 8T

Характеристики	**Xiaomi Redmi Note 8T**
Процесор	Qualcomm Snapdragon 665
RAM (оперативна пам’ять)	4GB
Операційна система	Android 9.0 (P)
Графічний процесор	Adreno 610
Основна камера	8Мп

4.2. Сканер QR-коду

Для ідентифікації моделі печатної плати у Web-системі користувачу необхідно відсканувати QR-код, розміщений безпосередньо на платі за допомогою камери пристрою. Скріншот роботи QR-сканеру представлено на рисунку 4.1.

Рисунок 4.1 – Сканер QR-коду

4.3. Відображення 3D-моделі

Після сканування QR-коду системою здійснюється запит про пошук до бази даних. Результатом успішного запиту буде поява повідомлення “Завантаження моделей” у нижній частині Web-сторінки (рисунок 4.2).

Рисунок 4.2 – Завантаження моделей

Результатом успішного завантаження 3D-моделі плати стане поява на екрані пристрою її тривимірного прототипу (рисунок 4.3).

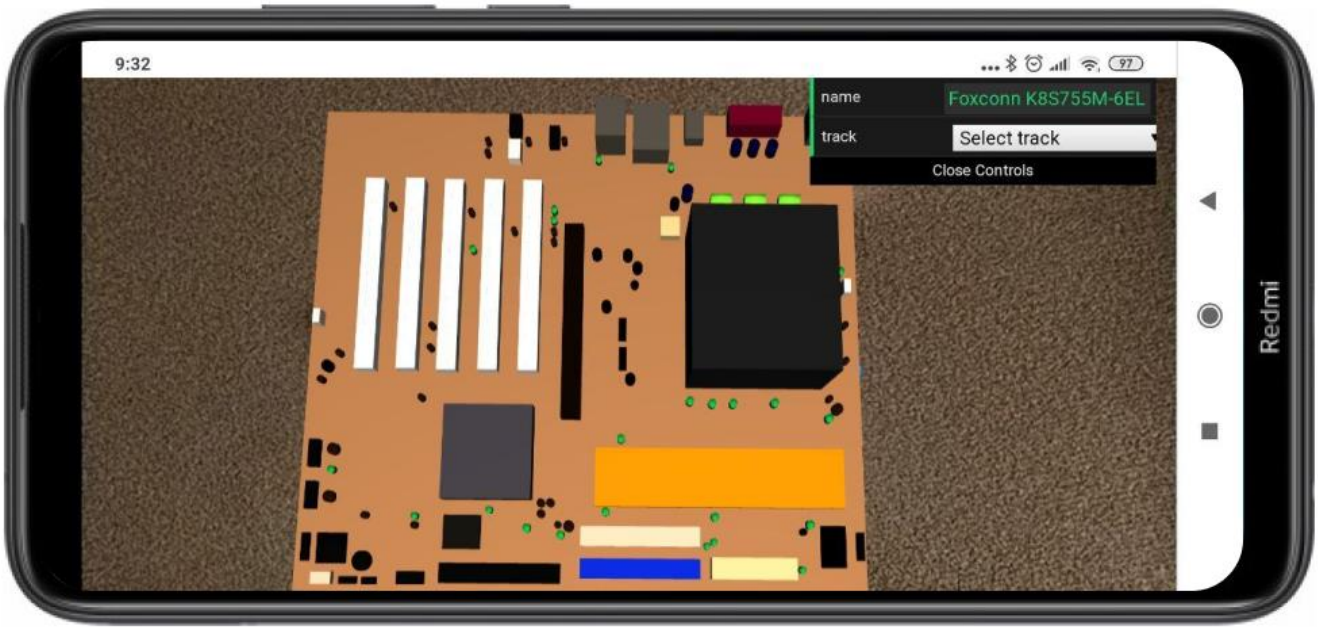

Рисунок 4.3 – Відображення 3D-моделі печатної плати на екрані пристрою користувача

4.4. Головне меню Web-застосунку

У правому верхньому куті Web-сторінки розташоване головне меню застосунку (рисунок 4.4).

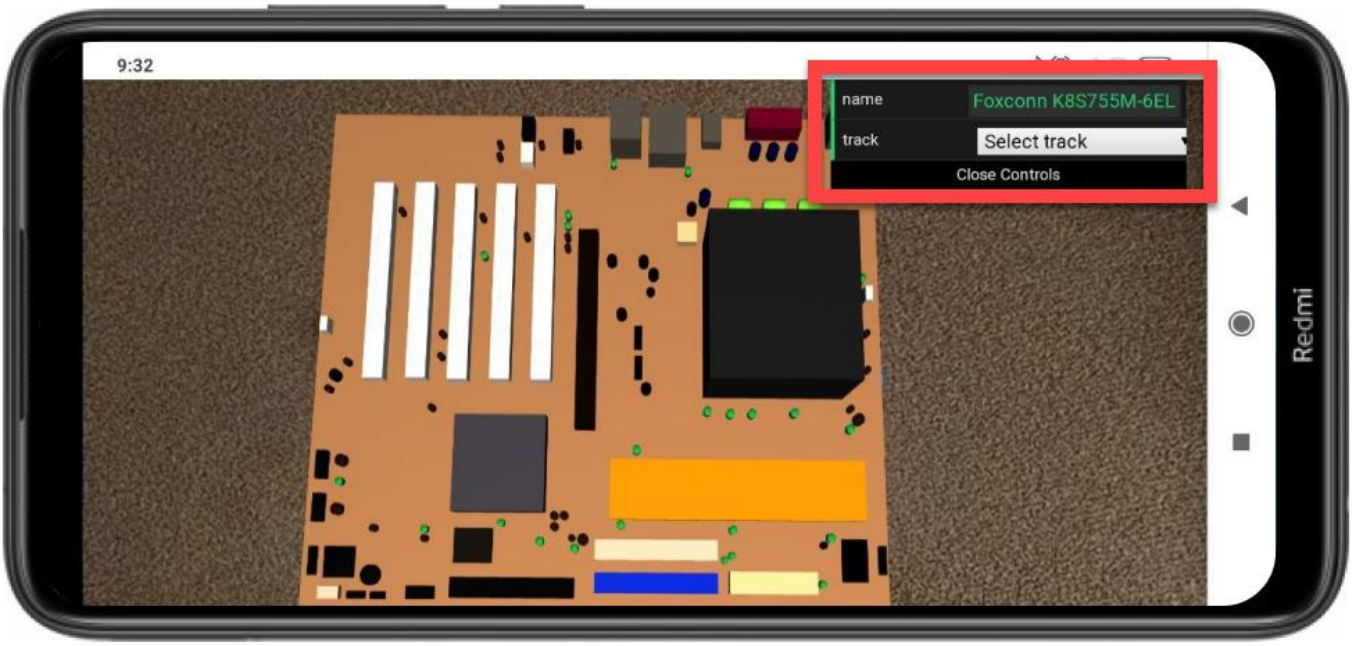

Рисунок 4.4 – Головне меню Web-застосунку

Головне меню має наступну структуру:

- поле “name”;
- розкривний список “track”;
- кнопку “Close Controls”.

У полі “name” знаходиться інформація про повне ім’я завантаженої моделі та її серійний номер. Це дає змогу користувачеві переконатись у тому, що було завантажено необхідну 3D-плату.

У розкривному списку “track” міститься інформація про існуючі струмопровідні доріжки завантаженої моделі плати. За замовчуванням Web-система

не відображає жодну доріжку на 3D-моделі. Доступ до розкривного списку існуючих доріжок здійснюється шляхом натискання користувачем на поле “Select track”.

Результатом натискання на поле “Select track” буде поява списку наявних струмопровідних доріжок на Web-сторінці (рисунок 4.5).

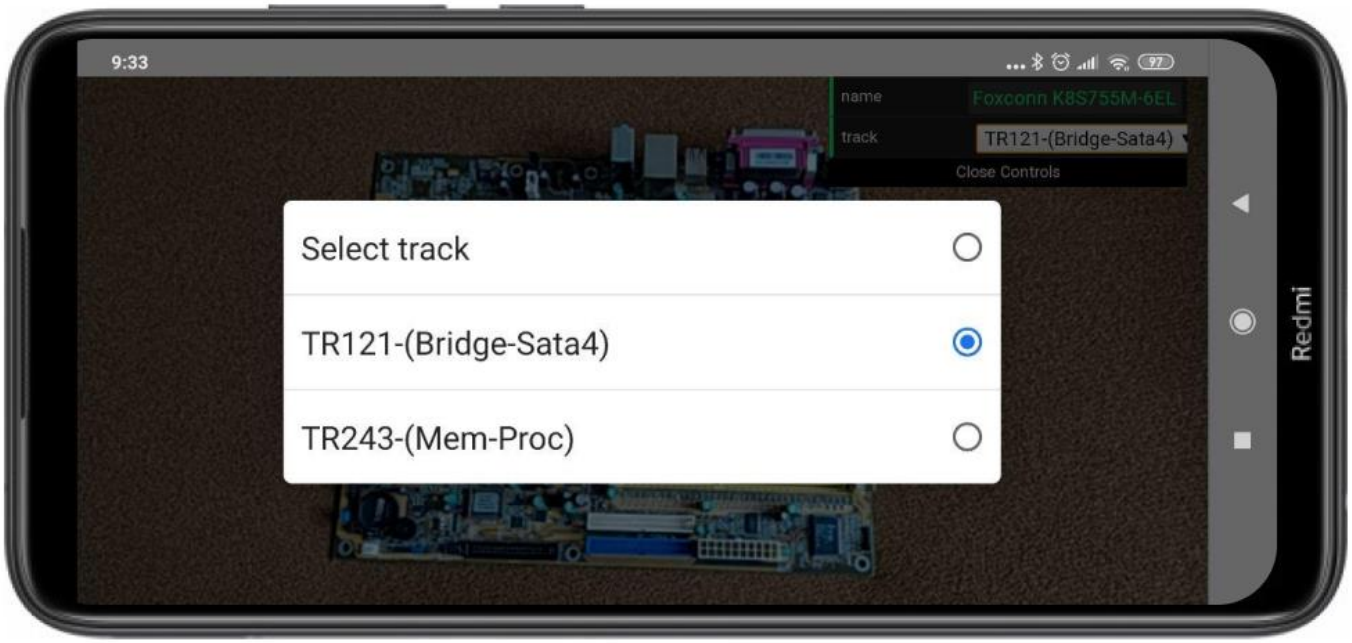

Рисунок 4.5 – Список наявних струмопровідних доріжок

Назва доріжки складається з її ідентифікаційного номеру (TR121, TR243) та короткого опису в дужках, що пояснює які компоненти плани вона поєднує (Bridge-Sata4 – південний міст та роз’єм Sata4, Mem-Proc – пам’ять та процесор). Приклади відображення струмопровідних доріжок TR121 та TR243 представлено на рисунках 4.6 та 4.7 відповідно.

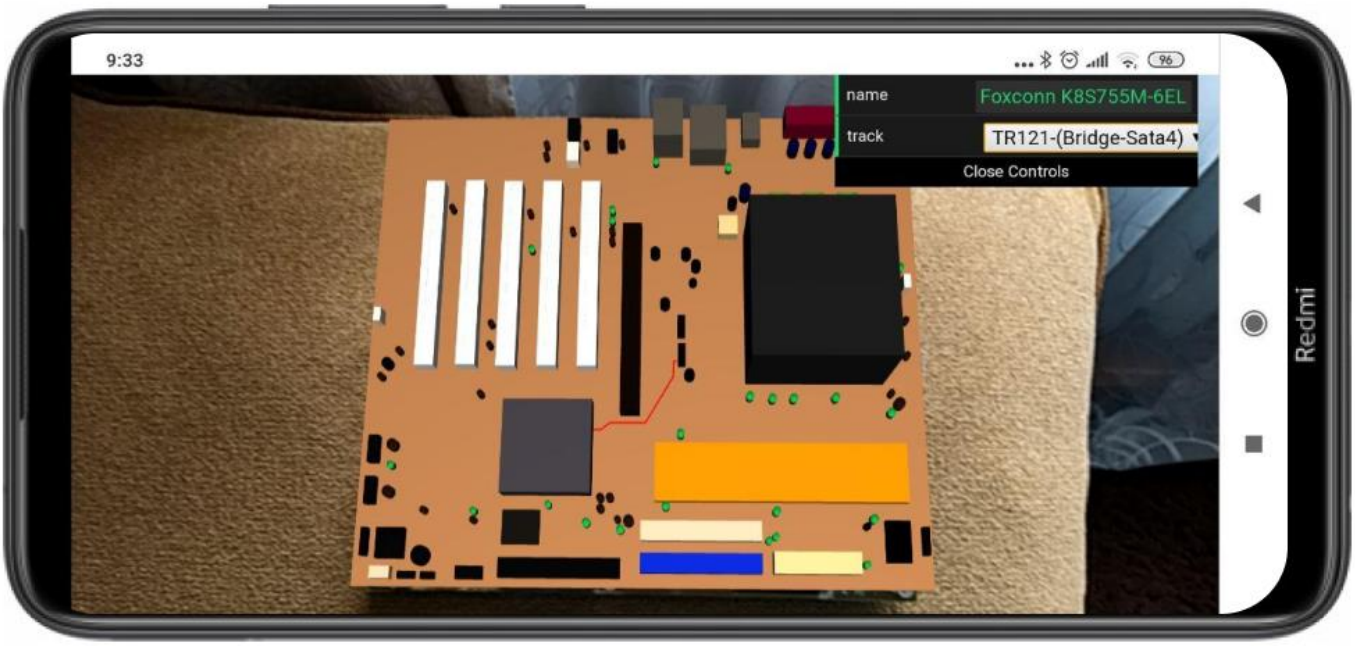

Рисунок 4.6 – Відображення доріжки TR121

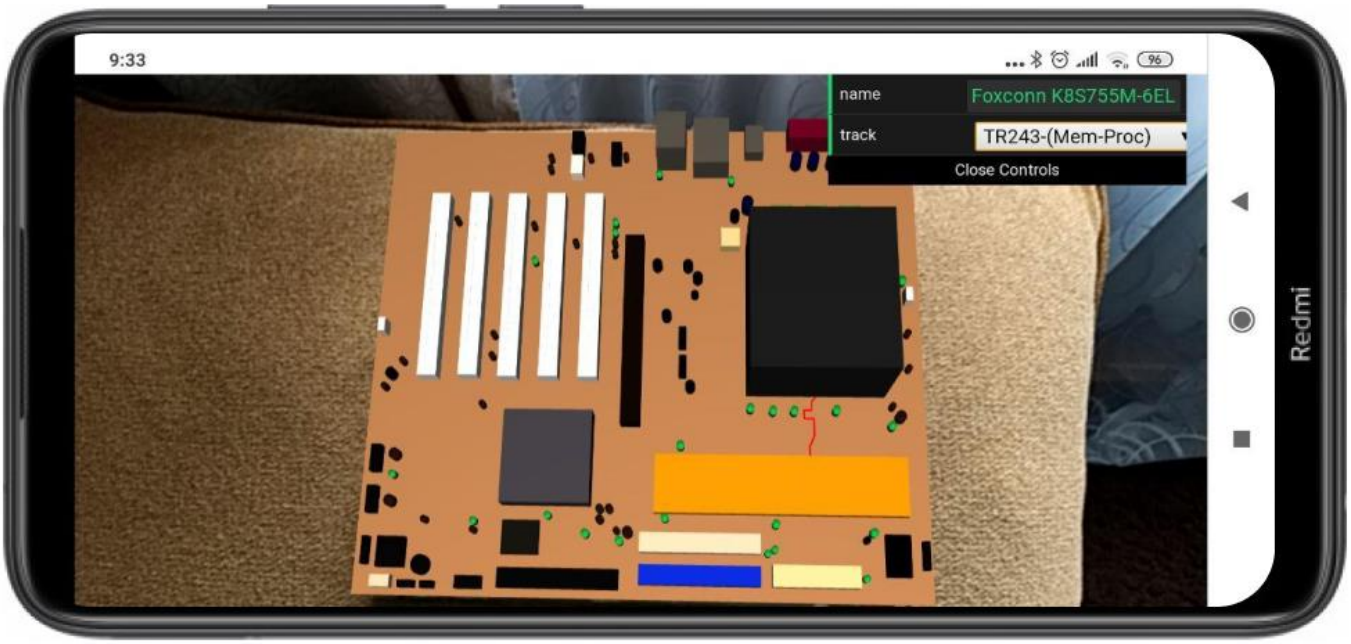

Рисунок 4.7 – Відображення доріжки TR243

Кнопка “Close Controls” відповідає за розгортання та згортання головного меню Web-застосунку для зручності користування (рисунок 4.8)

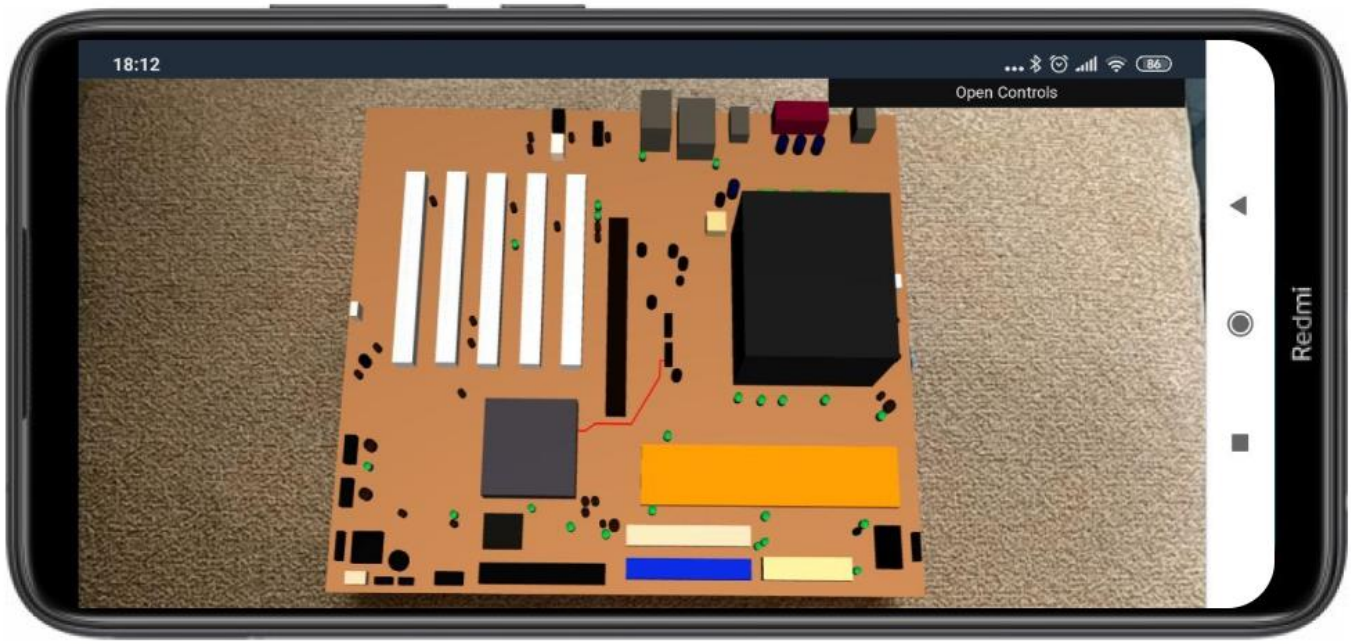

Рисунок 4.8 – Робота кнопки “Close Controls”

Користувальницький інтерфейс досить лаконічний та інтуїтивно-зрозумілий. Це пов’язано із тим, що при розробці Web-застосунку акцент був зроблений швидкість обробки даних. Відсутність великої кількості функціональних кнопок спрощує процес взаємодії користувача із Web-системою.

Висновки до розділу

У даному розділі наукової робити детально описано методику роботи користувача із програмною системою для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності.

Було визначено необхідні системні вимоги для коректної роботи Web-застосунку: швидке з’єднання із мережею Internet; наявність сучасного Web-браузера

Google Chrome версії 81.0.4044.138 і вище або Mozilla Firefox версії 76.0.1 і вище; об’єм оперативної пам’яті пристрою не менше 2GB та пристрій із камерою.

Окрім того, описано технічні характеристики пристроїв, на яких Web-застосунок успішно пройшов випробування.

ВИСНОВКИ

Дана робота присвячена використанню технології доповненої реальності для оптимізації сервісу печатних плат. Під час вирішення поставлених задач було отримано наступні результати:

1. Аналіз існуючих програмних продуктів для сервісу печатних плат показав, що наявні системи мають обмежену цільову аудиторію та високий поріг входження до сфери сервісу комп'ютерних складових. Показано, що використання технології доповненої реальності дає можливість зробити сервіс печатних плат значно ефективнішим.
2. Порівняльний аналіз сучасних технологій програмування для розробки Web-застосунків із використанням доповненої реальності показав, що найбільш відповідними засобами для реалізацій front-end частини продукту є JavaScript, Html5 та бібліотеки Three.js і AR.js, а для back-end частини – Node.js та Webpack. Використання зазначених вище технологій дозволяє реалізувати продукт із швидкою обробкою даних та інтуїтивно-зрозумілим інтерфейсом.
3. Розроблено модульну архітектуру застосунку, що дало змогу організувати ресурси для роботи проекту. Проведено дослідження залежності максимальної відстані ідентифікації маркеру від його розмірів, що дозволило обрати оптимальний розмір маркеру.
4. Реалізовано Web-застосунок, що дає змогу зробити процес ремонту печатних плат більш продуктивним за рахунок зменшення часових витрат на дослідження компонентів плати та з'єднань між ними. Також, процес навчання нових спеціалістів спрощується, завдяки наочному відображенню елементів плати.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Сайт Thinkmobiles // What is Augmented Reality (AR) and How does it work [Електронний ресурс] – Режим доступу: https://thinkmobiles.com/blog/what-is-augmented-reality/.
2. ГОСТ Р 53386 – 2009. Платы печатные. Термины и определения. М.: Стандартинформ, 2009, 16 с.
3. Сайт Inspectar // Accelerate Electronics Development [Електронний ресурс] – Режим доступу: https://www.inspectar.com/.
4. Сайт Nodejs // About [Електронний ресурс] – Режим доступу: https://nodejs.org/en/about/’.
5. Шелли П. Изучаем Node. Переходим на сторону сервера. 2-е изд. дополненное и переработанное / Пауэрс Шелли., 2016. – 304 с. – (Издательский дом "Питер").
6. Johnson B. Visual Studio Code: End-to-End Editing and Debugging Tools for Web Developers / Bruce Johnson. – Indianapolis, Indiana: Wiley, 2019. – 192 с. – (John Wiley & Sons). – (ISBN1119588219).
7. Гради Б. Язык UML. Руководство пользователя. 2-е изд. / Б. Гради, Д. Рамбо, И. Якобсон. – Москва: Академия Айти, 2006. – 496 с. – (ДМК Пресс).

ДОДАТОК А

Застосунок для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності

Специфікація

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61122 20Б

Аркушів 2

Київ 2020

Позначення	Найменування	Примітки
Документація		
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61122 20Б 81-1	Записка.docx	Текстова частина дипломної роботи
Компоненти		
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61122 20Б	main.js	Головний компонент застосунку
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61122 20Б	index.html	Файл із клієнтською частиною застосунку
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61122 20Б	main.css	Файл зі стилями клієнтської частини застосунку
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61122 20Б	*.obj	Файл із 3D-об’єктом
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61122 20Б	*.mtl	Файл із текстурою 3D-об’єкту

ДОДАТОК Б

Застосунок для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності

Текст програми

УКР.НТУУ“КПІ імені Ігоря Сікорського”_ТЕФ_АПЕПС_ТР61122_20Б

Аркушів 10

Київ 2020

Текст головного компоненту застосунку

```
import '@/assets/styles/main.css';
import jsQR from 'jsqr';
import {
  getBoard,
  getModelPath,
  getPatternPath,
  getAssetPath
} from '@/utils/index';
import Message from '@/utils/message';
import * as dat from 'dat.gui';
import { OBJLoader } from './utils/objloader';
import { MTLLoader } from './utils/mtlloader';

//showing tracks
const setTrack = (levelName) => {
  const sceneObj = objects.find((level) => level.name == currentLevel);
  if (sceneObj !== undefined) {
    if (!sceneObj.isBase) {
      const sceneObj = scene.getObjectByName(currentLevel);
      if (sceneObj) {
        scene.remove(sceneObj);
      }
    }
  }

  const nextSceneObj = objects.find((level) => level.name == levelName);

  if (nextSceneObj !== undefined) {
    scene.add(nextSceneObj.obj);
    currentLevel = levelName;
  } else {
    currentLevel = '';
  }
};
//objects load
const loadObject = (
  boardId,
```

```
  level,
  { material, model, scale, position },
  index
) => {
  new MTLLoader(manager).load(getAssetPath(material), function (materials) {
    materials.preload();

    new OBJLoader(manager)
      .setMaterials(materials)
      .load(getAssetPath(model), function (obj) {
        decreaseRemainedAmount(boardId);

        obj.scale.set(...scale);
        obj.position.set(...position);
        obj.name = level.name;

        objects.push({
          boardId,
          name: level.name,
          isBase: index == 0,
          obj
        });

        if (index == 0) {
          currentLevel = level.name;
          scene.add(obj);
        }
      });
  });
};

function decreaseRemainedAmount(boardId) {
  remainedObjects--;
  if (remainedObjects == 0) {
    message.hide();
    initTrackController(boardId);
  }
}
```

```
//init pcb
function initTrackController(boardId) {
  modelController.track = '';

  const board = boards.find((board) => board.id == boardId);

  const placeholderText = 'Select track';

  let trackController = gui.add(
    modelController,
    'track',
    board.levels.map((level, index) =>
      index == 0 ? placeholderText : level.name
    )
  );

  trackController.setValue(placeholderText).onChange(setTrack);
}

const message = new Message();

var boards = [],
  objects = [],
  remainedObjects = 0;

var video, canvas, context;

var modelController, gui, currentLevel;

var renderer = new THREE.WebGLRenderer({
  preserveDrawingBuffer: true,
  antialias: true,
  alpha: true,
  colorManagment: true,
  logarithmicDepthBuffer: true,
  sortObjects: true
});
renderer.setPixelRatio(window.devicePixelRatio);
renderer.setSize(window.innerWidth, window.innerHeight);
```

```
renderer.setClearColor(new THREE.Color('lightgrey'), 0);
renderer.domElement.style.position = 'absolute';
renderer.domElement.style.top = '0px';
renderer.domElement.style.left = '0px';
document.body.appendChild(renderer.domElement);

message.show('Наведіть камеру на QR код');

var onRenderFcts = [];

//scene
var scene = new THREE.Scene();

//camera
var camera = new THREE.PerspectiveCamera(45, 2, 0.01, 100000);
scene.add(camera);

//light
var light = new THREE.PointLight(0xffffff, 1.44);
light.position.set(0, 15, 7);

scene.add(light);

var arToolkitSource = new THREEx.ArToolkitSource({
  sourceType: 'webcam'
});

arToolkitSource.init(function onReady() {
  setTimeout(() => {
    onResize();

    video = document.querySelector('video');

    canvas = document.createElement('canvas');
    context = canvas.getContext('2d');
  }, 2000);
});

window.addEventListener('resize', function () {
```

```
  onResize();
});

//resize window
function onResize() {
  arToolkitSource.onResizeElement();
  arToolkitSource.copyElementSizeTo(renderer.domElement);
  if (arToolkitContext.arController !== null) {
    arToolkitSource.copyElementSizeTo(arToolkitContext.arController.canvas);
  }
}

var arToolkitContext = new THREEx.ArToolkitContext({
  detectionMode: 'mono'
});

arToolkitContext.init(function onCompleted() {
  camera.projectionMatrix.copy(arToolkitContext.getProjectionMatrix());
});

onRenderFcts.push(function () {
  if (arToolkitSource.ready === false) return;

  arToolkitContext.update(arToolkitSource.domElement);

  scene.visible = camera.visible;
});

scene.visible = false;

const manager = new THREE.LoadingManager();

onRenderFcts.push(function () {
  renderer.render(scene, camera);
});

//render
onRenderFcts.push(function () {
  if (video && boards.length == 0) {
```

```
    if (video.readyState === video.HAVE_ENOUGH_DATA) {
      canvas.hidden = false;

      canvas.height = video.videoHeight;
      canvas.width = video.videoWidth;

      context.drawImage(video, 0, 0, canvas.width, canvas.height);
      var imageData = context.getImageData(0, 0, canvas.width, canvas.height);

      var code = jsQR(imageData.data, imageData.width, imageData.height, {
        inversionAttempts: 'dontInvert'
      });

      if (code && code.data) {
        if (boards.find((board) => board.id == code.data) == undefined) {
          initBoard(code.data);
        }
      }
    }
  }
});

var lastTimeMsec = null;
requestAnimationFrame(function animate(nowMsec) {
  requestAnimationFrame(animate);
  lastTimeMsec = lastTimeMsec || nowMsec - 1000 / 60;
  var deltaMsec = Math.min(200, nowMsec - lastTimeMsec);
  lastTimeMsec = nowMsec;

  onRenderFcts.forEach(function (onRenderFct) {
    onRenderFct(deltaMsec / 1000, nowMsec / 1000);
  });
});

function initBoard(id) {
  message.hide();
  const board = getBoard(id);
  boards.push(board);
  console.log(`Found board ${id}`);
```

```
  if (boards.length == 1) {
    modelController = {
      name: board.name
    };
    gui = new dat.GUI();
    gui.add(modelController, 'name');

    let modelParams = {
      scale: board.scale || [1, 1, 1],
      position: board.position || [0, 0, 0]
    };
    remainedObjects = board.levels.length;

    message.show('Завантаження моделей');

    board.levels.map((level, index) => {
      let modelPath = getModelPath(board.id, level.file);
      let params = { ...modelPath, ...modelParams };
      loadObject(id, level, params, index);
    });
  }
  let patternPath = getPatternPath(board.id);
  //marker controls
  var markerControls = new THREEx.ArMarkerControls(arToolkitContext, camera, {
    type: 'pattern',
    // patternUrl: patternPath,
    patternUrl: 'patterns/hiro.patt',
    changeMatrixMode: 'cameraTransformMatrix',
    smooth: true,
    smoothcount: 4,
    smoothTolerance: 0.01,
    smoothThreshold: 2
  });
}
```

Текст файлу package.json із вказаними залежностями

```
{
  "name": "motherboard",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "private": true,
  "scripts": {
    "build": "webpack --env.NODE_ENV=production",
    "dev": "webpack-dev-server --host 0.0.0.0 --env.NODE_ENV=local"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "devDependencies": {
    "@babel/core": "^7.8.3",
    "@babel/preset-env": "^7.8.3",
    "babel-loader": "^8.0.6",
    "css-loader": "^3.4.2",
    "file-loader": "^5.0.2",
    "style-loader": "^1.1.3",
    "webpack": "^4.41.5",
    "webpack-cli": "^3.3.10",
    "webpack-dev-server": "^3.10.1"
  },
  "dependencies": {
    "dat.gui": "^0.7.7",
    "jsqr": "^1.2.0"
  }
}
```

Текст файлу webpack.config із вказаними залежностями

```
const path = require('path');
const webpack = require('webpack');
module.exports = (env) => {
  return {
    entry: './src/index.js',
    devtool: 'inline-source-map',
    output: {
      filename: 'main.js',
      path: path.resolve(__dirname, 'dist')
    },
    devServer: {
      contentBase: path.join(__dirname, 'dist'),
      compress: true,
      port: 8080
    },
    resolve: {
      extensions: ['.js', '.vue', '.json'],
      alias: {
        '@': path.resolve('src')
      }
    },
    module: {
      rules: [
        {
          test: /\.js$/,
          exclude: /(node_modules)/,
          use: {
            loader: 'babel-loader'
          }
        },
        {
          test: /\.css$/,
          use: [
            'style-loader',
            {
              loader: 'css-loader',
              options: {
```

```
              modules: false
            }
          }
        ]
      },
      {
        test: /\.(png|svg|jpg)$/,
        use: ['file-loader']
      }
    ]
  },
  plugins: [
    new webpack.DefinePlugin({
      BASE_DIR: JSON.stringify(env.NODE_ENV == 'local' ? '' : '/pcb_ar/dist')
    })
  ]
};
};
```

ДОДАТОК В

Застосунок для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності

Опис програмного коду

УКР.НТУУ“КПІ імені Ігоря Сікорського”_ТЕФ_АПЕПС_ТР61122_20Б

Аркушів 9

Київ 2020

АНОТАЦІЯ

Даний додаток містить опис застосунку для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності. Реалізований програмний продукт виконує наступні завдання:

- зчитування та розпізнавання QR-кодів;
- зчитування та розпізнавання Hiro-маркерів;
- відображення 3D-моделі печатної плати на екрані пристрою із камерою у режимі реального часу;
- вибір та відображення струмопровідних доріжок, що з’єднують певні компоненти плати на 3D-моделі.

Web-застосунок було реалізовано за допомогою технологій JavaScript, Html5, CSS, бібліотек Three.js і AR.js, середовища виконання Node.js, інструменту для компіляції модулів Webpack у середовищі виконання Visual Studio Code.

ЗМІСТ

1. ЗАГАЛЬНІ ВІДОМОСТІ

У цьому додатку наведено опис застосунку для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності. У додатку Б наведено програмний код головних компонентів розробленої системи.

Для роботи із реалізованою системою необхідно мати швидке з’єднання із мережею Internet, сучасний Web-браузер Google Chrome версії 81.0.4044.138 і вище або Mozilla Firefox версії 76.0.1 і вище та пристрій із камерою (смартфон, планшет, портативний або персональний комп’ютер).

При розробці Web-застосунку використовувались: середовище виконання Visual Studio Code, мова програмування JavaScript, бібліотеки Three.js і AR.js, середовище виконання Node.js та інструмент для компіляції модулів Webpack.

2. ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Розроблені компоненти застосунку виконують завдання зчитування та розпізнавання QR-кодів, зчитування та розпізнавання Hiro-маркерів, відображення 3D-моделі печатної плати на екрані пристрою із камерою у режимі реального часу та вибір струмопровідних доріжок, що з’єднують певні компоненти плати на 3D-моделі.

Призначенням створеного програмного забезпечення є наочне відображення компонентів печатної плати та відповідних струмопровідних доріжок, що їх з’єднують. Потенційними користувачами даного застосунку стануть власники комп’ютерних ремонтних сервісів.

3. ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Згідно з архітектурою, Web-застосунок складається із трьох головних модулів:

- QR-модулю;
- файлової бази даних моделей;
- AR-модулю.

За допомогою камери пристрою QR-модуль розпізнає QR-код на печатній платі та ідентифікує її. Далі QR-модуль надсилає пошуковий запит до бази даних.

Файлова база даних містить у собі інформацію про печатні плати та їх струмопровідні доріжки.

AR-модуль отримує результат пошуку та розпізнає у відеопотоці Hiro-маркер. Внаслідок успішної ідентифікації маркера модуль доповненої реальності відображає відповідну 3D-модель безпосередньо над маркером. Таким чином, у режимі реального часу на екрані пристрою з’являється 3D-модель печатної плати.

4. ВИКОРИСТОВУВАНІ ТЕХНІЧНІ ЗАСОБИ

Для роботи із Web-застосунком для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності не потрібно інсталювати стороннього програмного забезпечення, адже реалізована система є Web-системою.

Обов’язкові вимоги для роботи застосунку наступні:

1. Швидке з’єднання із мережею Internet (не менше 5Мбіт/с).
2. Наявність сучасного Web-браузера Google Chrome версії 81.0.4044.138 і вище або Mozilla Firefox версії 76.0.1 і вище.
3. Пристрій із камерою (смартфон, планшет, портативний або персональний комп’ютер).
4. Об’єм оперативної пам’яті пристрою не менше 2GB.

5. ВИКЛИК І ЗАВАНТАЖЕННЯ

Реалізований Web-застосунок для візуалізації вказівок із сервісу печатних плат на основі доповненої реальності не потребує інсталяції, адже є Web-системою.

Доступ до системи здійснюється за URL-посиланням https://failcatcher.github.io/pcb_ar/dist/.

6. ВХІДНІ ДАНІ

Вхідними даними для розробленого Web-застосунку є QR-код та Hiro-маркер для ідентифікації моделі печатної плати.

7. ВИХІДНІ ДАНІ

Вихідними даними розробленого Web-застосунку є відображення 3D-моделі печатної плати на екрані пристрою із камерою.